```
void __attribute__((naked)) task_switch()
{
    asm volatile(
    "    addi    sp,sp,-60       \n"
    "    sw      x1,0(sp)        \n"
    "    sw      x3,4(sp)        \n"
    "    sw      x4,8(sp)        \n"
    "    sw      x5,12(sp)       \n"
    "    sw      x6,16(sp)       \n"
    "    sw      x7,20(sp)       \n"
    "    sw      x8,24(sp)       \n"
    "    sw      x9,28(sp)       \n"
    "    sw      x10,32(sp)      \n"
    "    sw      x11,36(sp)      \n"
    "    sw      x12,40(sp)      \n"
    "    sw      x13,44(sp)      \n"
    "    sw      x14,48(sp)      \n"
    "    sw      x15,52(sp)      \n"
    "    csrrw   x1,0x341,zero   \n" // Interrupted PC
    "    sw      x1,56(sp)       \n"
    "    mv      a0,sp           \n"
    "    call    getnextsp       \n"
    "    mv      sp,a0           \n"
    "    lw      x1,56(sp)       \n"
    "    csrrw   zero,0x341,x1   \n" // Interrupted PC
    "    lw      x1,0(sp)        \n"
    "    lw      x3,4(sp)        \n"
    "    lw      x4,8(sp)        \n"
    "    lw      x5,12(sp)       \n"
    "    lw      x6,16(sp)       \n"
    "    lw      x7,20(sp)       \n"
    "    lw      x8,24(sp)       \n"
    "    lw      x9,28(sp)       \n"
    "    lw      x10,32(sp)      \n"
    "    lw      x11,36(sp)      \n"
    "    lw      x12,40(sp)      \n"
    "    lw      x13,44(sp)      \n"
    "    lw      x14,48(sp)      \n"
    "    lw      x15,52(sp)      \n"
    "    addi    sp,sp,60        \n"
    "    mret                    \n"
    );
}

// task table
#define MAXTSK 4
struct {
    uint32_t tskix;         // Current task index
    uint32_t tsksp[MAXTSK]; // list of SPs
} tsktab;

// Scheduler: round-robin: Saves SP and gets a new one fron the task list
uint32_t getnextsp(uint32_t sp)
{
    uint32_t i;
    i=tsktab.tskix;
    tsktab.tsksp[i]=sp;
    i++;
    if (!tsktab.tsksp[i]) i=0;  // Task list ends with SP=0
    sp=tsktab.tsksp[i];
    tsktab.tskix=i;
    return sp;
}

#define STKSZ0 (1<<10)  // Stack size for task 0

void init_task()
{
    uint32_t *sp;
    sp=(uint32_t *)(0x20000000+(128<<10)-STKSZ0);
    sp=&sp[-15];
    sp[14]=(uint32_t) tetris_task1; // PC of task #1 at its stack
    tsktab.tskix=0;                 // Current task number
    // task #0 is the current code and needs no initialization
    tsktab.tsksp[1]=(uint32_t)sp;   // SP of task #1
    tsktab.tsksp[2]=0;              // End of task list
    IRQVECT0=(uint32_t)task_switch; // trap vector
}
```